

# Package 'earthUI'

March 11, 2026

**Title** Interactive 'shiny' GUI for the 'earth' Package

**Version** 0.1.1

**Description** Provides a 'shiny'-based graphical user interface for the 'earth' package, enabling interactive building and exploration of Multivariate Adaptive Regression Splines (MARS) models. Features include data import from CSV and 'Excel' files, automatic detection of categorical variables, interactive control of interaction terms via an allowed matrix, comprehensive model diagnostics with variable importance and partial dependence plots, and publication-quality report generation via 'Quarto'.

**License** AGPL (>= 3)

**URL** <https://github.com/wcraytor/earthUI>

**BugReports** <https://github.com/wcraytor/earthUI/issues>

**Depends** R (>= 4.1.0)

**Imports** earth (>= 5.3.0), ggplot2, readxl, shiny, stats, tools, utils

**Suggests** bslib, callr, DBI, DT, jsonlite, knitr, plotly, quarto, rmarkdown, RSQLite, showtext, sysfonts, testthat (>= 3.0.0), writexl

**Config/testthat/edition** 3

**Encoding** UTF-8

**VignetteBuilder** knitr

**RoxygenNote** 7.3.3

**NeedsCompilation** no

**Author** William Craytor [aut, cre]

**Maintainer** William Craytor <bcraytor@proton.me>

**Repository** CRAN

**Date/Publication** 2026-03-11 16:50:14 UTC

## Contents

build_allowed_function . . . . .	2
build_allowed_matrix . . . . .	3
detect_categoricals . . . . .	4
detect_types . . . . .	4
fit_earth . . . . .	5
format_anova . . . . .	8
format_model_equation . . . . .	9
format_summary . . . . .	10
format_variable_importance . . . . .	11
import_data . . . . .	11
launch . . . . .	12
list_g_functions . . . . .	13
plot_actual_vs_predicted . . . . .	14
plot_contribution . . . . .	14
plot_correlation_matrix . . . . .	15
plot_g_contour . . . . .	16
plot_g_function . . . . .	16
plot_g_persp . . . . .	17
plot_partial_dependence . . . . .	18
plot_qq . . . . .	19
plot_residuals . . . . .	19
plot_variable_importance . . . . .	20
render_report . . . . .	21
validate_types . . . . .	21
<b>Index</b>	<b>23</b>

---

build\_allowed\_function

*Build an allowed function for earth()*

---

### Description

Converts an allowed interaction matrix into a function compatible with the allowed parameter of `earth::earth()`. The function checks that ALL pairwise combinations among the predictors in a proposed interaction term are TRUE in the matrix.

### Usage

```
build_allowed_function(allowed_matrix)
```

### Arguments

`allowed_matrix` A symmetric logical matrix as returned by `build_allowed_matrix()`.

## Details

The returned function implements the standard `earth()` allowed function contract. When `earth` proposes a new hinge function involving predictor `pred` with existing parent predictors indicated by the parents logical vector, the function checks that every pair of involved predictors is allowed in the matrix.

For a 3-way interaction between `X`, `Y`, `Z`, the function verifies that `(X,Y)`, `(Y,Z)`, and `(X,Z)` are all `TRUE` in the matrix.

## Value

A function with signature `function(degree, pred, parents, namesx, first)` suitable for the allowed parameter of `earth::earth()`.

## Examples

```
mat <- build_allowed_matrix(c("sqft", "bedrooms", "pool"))
mat["sqft", "pool"] <- FALSE
mat["pool", "sqft"] <- FALSE
func <- build_allowed_function(mat)
```

---

`build_allowed_matrix` *Build an allowed interaction matrix*

---

## Description

Creates a symmetric logical matrix indicating which pairs of predictors are allowed to interact. By default, all interactions are allowed.

## Usage

```
build_allowed_matrix(variable_names, default = TRUE)
```

## Arguments

`variable_names` Character vector of predictor variable names.

`default` Logical. Default value for all entries. Default is `TRUE` (all interactions allowed).

## Value

A symmetric logical matrix with `variable_names` as both row and column names.

## Examples

```
mat <- build_allowed_matrix(c("sqft", "bedrooms", "pool"))
mat["sqft", "pool"] <- FALSE
mat["pool", "sqft"] <- FALSE
mat
```

---

detect\_categoricals     *Detect likely categorical variables in a data frame*

---

### Description

Returns a logical named vector indicating which columns are likely categorical. Character and factor columns are always flagged. Numeric columns with fewer than `max_unique` unique values are also flagged.

### Usage

```
detect_categoricals(df, max_unique = 10L)
```

### Arguments

`df`                     A data frame.  
`max_unique`            Integer. Numeric columns with this many or fewer unique values are flagged as likely categorical. Default is 10.

### Value

A named logical vector with one element per column. TRUE indicates the column is likely categorical.

### Examples

```
df <- data.frame(  
  price = c(100, 200, 300, 400),  
  pool = c("Y", "N", "Y", "N"),  
  bedrooms = c(2, 3, 2, 4),  
  sqft = c(1200, 1500, 1300, 1800)  
)  
detect_categoricals(df)
```

---

detect\_types             *Detect column types in a data frame*

---

### Description

Inspects each column and returns a best-guess R type string. Character columns are tested for common date patterns. Numeric columns containing only 0/1 values (with both present) are flagged as logical.

### Usage

```
detect_types(df)
```

**Arguments**

df                    A data frame.

**Value**

A named character vector with one element per column. Possible values: "numeric", "integer", "character", "logical", "factor", "Date", "POSIXct", "unknown".

**Examples**

```
df <- data.frame(
  price = c(100.5, 200.3, 300.1),
  rooms = c(2L, 3L, 4L),
  pool = c("Y", "N", "Y"),
  sold = c(TRUE, FALSE, TRUE)
)
detect_types(df)
```

---

fit\_earth

*Fit an earth model*

---

**Description**

Wrapper around `earth::earth()` with parameter validation and automatic cross-validation when interaction terms are enabled.

**Usage**

```
fit_earth(
  df,
  target,
  predictors,
  categoricals = NULL,
  linpreds = NULL,
  type_map = NULL,
  degree = 1L,
  allowed_func = NULL,
  allowed_matrix = NULL,
  nfold = NULL,
  nprune = NULL,
  thresh = NULL,
  penalty = NULL,
  minspan = NULL,
  endspan = NULL,
  fast.k = NULL,
  pmethod = NULL,
  glm = NULL,
```

```

trace = NULL,
nk = NULL,
newvar.penalty = NULL,
fast.beta = NULL,
ncross = NULL,
stratify = NULL,
varmod.method = NULL,
varmod.exponent = NULL,
varmod.conv = NULL,
varmod.clamp = NULL,
varmod.minspan = NULL,
keepxy = NULL,
Scale.y = NULL,
Adjust.endspan = NULL,
Auto.linpreds = NULL,
Force.weights = NULL,
Use.beta.cache = NULL,
Force.xtx.prune = NULL,
Get.leverages = NULL,
Exhaustive.tol = NULL,
wp = NULL,
weights = NULL,
...,
.capture_trace = TRUE
)

```

### Arguments

df	A data frame containing the modeling data.
target	Character string. Name of the response variable.
predictors	Character vector. Names of predictor variables.
categoricals	Character vector. Names of predictors to treat as categorical (converted to factors before fitting). Default is NULL.
linpreds	Character vector. Names of predictors constrained to enter the model linearly (no hinge functions). Default is NULL.
type_map	Named list or character vector. Maps column names to declared types (e.g., "numeric", "Date", "factor"). When provided, columns are coerced before fitting: Date/POSIXct columns are converted to numeric (days/seconds since epoch), and type-derived categoricals are merged into categoricals. Default is NULL (no coercion).
degree	Integer. Maximum degree of interaction. Default is 1 (no interactions). When $\geq 2$ , cross-validation is automatically enabled.
allowed_func	Function or NULL. An allowed function as returned by <code>build_allowed_function()</code> . Only used when degree $\geq 2$ .
allowed_matrix	Logical matrix or NULL. The allowed interaction matrix. Stored in the result for report export. Not used for fitting (use <code>allowed_func</code> instead).

nfold	Integer. Number of cross-validation folds. Automatically set to 10 when degree $\geq 2$ unless explicitly provided. Set to 0 to disable.
nprune	Integer or NULL. Maximum number of terms in the pruned model.
thresh	Numeric. Forward stepping threshold. Default is earth's default (0.001).
penalty	Numeric. Generalized cross-validation penalty per knot. Default is earth's default (if degree $> 1, 3$ ; otherwise 2).
minspan	Integer or NULL. Minimum number of observations between knots.
endspan	Integer or NULL. Minimum number of observations before the first and after the last knot.
fast.k	Integer. Maximum number of parent terms considered at each step of the forward pass. Default is earth's default (20).
pmethod	Character. Pruning method. One of "backward", "none", "exhaustive", "forward", "seqrep", "cv". Default is "backward".
glm	List or NULL. If provided, passed to earth's glm argument to fit a GLM on the earth basis functions.
trace	Numeric. Trace earth's execution. 0 (default) = none, 0.3 = variance model, 0.5 = cross validation, 1-5 = increasing detail.
nk	Integer or NULL. Maximum number of model terms before pruning.
newvar.penalty	Numeric or NULL. Penalty for adding a new variable in the forward pass (Friedman's gamma). Default 0.
fast.beta	Numeric or NULL. Fast MARS ageing coefficient. Default 1.
ncross	Integer or NULL. Number of cross-validations. Default 1.
stratify	Logical or NULL. Stratify cross-validation samples. Default TRUE.
varmod.method	Character or NULL. Variance model method. One of "none", "const", "lm", "rlm", "earth", "gam", "power", "power0", "x.lm", "x.rlm", "x.earth", "x.gam".
varmod.exponent	Numeric or NULL. Power transform for variance model.
varmod.conv	Numeric or NULL. Convergence criterion for IRLS.
varmod.clamp	Numeric or NULL. Minimum estimated standard deviation.
varmod.minspan	Integer or NULL. minspan for internal variance model.
keepxy	Logical or NULL. Retain x, y in model object. Default FALSE.
Scale.y	Logical or NULL. Scale response internally. Default TRUE.
Adjust.endspan	Numeric or NULL. Interaction ends span multiplier. Default 2.
Auto.linpreds	Logical or NULL. Auto-detect linear predictors. Default TRUE.
Force.weights	Logical or NULL. Force weighted code path. Default FALSE.
Use.beta.cache	Logical or NULL. Cache coefficients in forward pass. Default TRUE.
Force.xtx.prune	Logical or NULL. Force X'X-based pruning. Default FALSE.
Get.leverages	Logical or NULL. Calculate hat values. Default TRUE.

Exhaustive.to1	Numeric or NULL. Condition number threshold for exhaustive pruning. Default 1e-10.
wp	Numeric vector or NULL. Response weights.
weights	Numeric vector or NULL. Case weights passed to earth.
...	Additional arguments passed to <code>earth::earth()</code> .
.capture_trace	Logical. If TRUE (default), capture earth's trace output. Set to FALSE when running in a background process.

### Value

A list with class "earthUI\_result" containing:

- model** The fitted earth model object.
- target** Name of the response variable.
- predictors** Names of predictor variables used.
- categoricals** Names of categorical predictors.
- degree** Degree of interaction used.
- cv\_enabled** Logical; whether cross-validation was used.
- data** The data frame used for fitting.

### Examples

```
df <- mtcars
result <- fit_earth(df, target = "mpg",
                   predictors = c("cyl", "disp", "hp", "wt"))
format_summary(result)
```

---

format_anova	<i>Format ANOVA decomposition</i>
--------------	-----------------------------------

---

### Description

Extracts the ANOVA table from a fitted earth model.

### Usage

```
format_anova(earth_result)
```

### Arguments

earth\_result An object of class "earthUI\_result" as returned by `fit_earth()`.

### Value

A data frame with the ANOVA decomposition showing which predictors contribute to each basis function and their importance.

## Examples

```
result <- fit_earth(mtcars, "mpg", c("cyl", "disp", "hp", "wt"))
format_anova(result)
```

---

format\_model\_equation *Format earth model as LaTeX equation*

---

## Description

Converts a fitted earth model into a LaTeX-formatted mathematical representation using g-function notation. Basis functions are grouped by degree (constant, first-degree, second-degree, third-degree) and labeled with indices that encode the group, position, and factor variable count.

## Usage

```
format_model_equation(earth_result, digits = 7L, response_idx = NULL)
```

## Arguments

earth_result	An object of class "earthUI_result" as returned by <code>fit_earth()</code> .
digits	Integer. Number of significant digits for coefficients and cut points. Default is 7.
response_idx	Integer or NULL. For multivariate models, which response column to generate the equation for (1-based). Default NULL returns all response equations in an earthUI_equation_multi object.

## Value

A list containing:

**latex** Character string. LaTeX array environment for HTML/MathJax rendering.

**latex\_inline** Character string. Wrapped in display math delimiters for MathJax/HTML rendering.

**latex\_pdf** Character string. LaTeX for native PDF output with escaped special characters in text blocks.

**latex\_word** Character string. LaTeX for Word/docx output.

**groups** List of group structures for programmatic access.

## Examples

```
result <- fit_earth(mtcars, "mpg", c("cyl", "disp", "hp", "wt"))
eq <- format_model_equation(result)
cat(eq$latex)
```

---

format_summary	<i>Format earth model summary</i>
----------------	-----------------------------------

---

### Description

Extracts key statistics from a fitted earth model including coefficients, basis functions, R-squared, GCV, GRSq, and RSS.

### Usage

```
format_summary(earth_result)
```

### Arguments

`earth_result` An object of class "earthUI\_result" as returned by `fit_earth()`.

### Value

A list containing:

**coefficients** Data frame of model coefficients and basis functions.

**r\_squared** Training R-squared.

**gcv** Generalized cross-validation value.

**grsq** Generalized R-squared (1 - GCV/variance).

**rss** Residual sum of squares.

**n\_terms** Number of terms in the pruned model.

**n\_predictors** Number of predictors used in the final model.

**n\_obs** Number of observations.

**cv\_rsq** Cross-validated R-squared (if CV was used, else NA).

### Examples

```
result <- fit_earth(mtcars, "mpg", c("cyl", "disp", "hp", "wt"))
summary_info <- format_summary(result)
summary_info$r_squared
```

---

```
format_variable_importance
      Format variable importance
```

---

**Description**

Extracts variable importance scores from a fitted earth model using `earth::evimp()`.

**Usage**

```
format_variable_importance(earth_result)
```

**Arguments**

`earth_result` An object of class "earthUI\_result" as returned by `fit_earth()`.

**Value**

A data frame with columns `variable`, `nsubsets`, `gcv`, and `rss`, sorted by overall importance (`nsubsets`).

**Examples**

```
result <- fit_earth(mtcars, "mpg", c("cyl", "disp", "hp", "wt"))
format_variable_importance(result)
```

---

```
import_data      Import data from CSV or Excel files
```

---

**Description**

Reads a CSV (.csv) or 'Excel' (.xlsx, .xls) file and returns a data frame. Column names are converted to snake\_case and duplicates are made unique.

**Usage**

```
import_data(filepath, sheet = 1, ...)
```

**Arguments**

`filepath` Character string. Path to the data file. Supported formats: .csv, .xlsx, .xls.  
`sheet` Character or integer. For Excel files, the sheet to read. Defaults to the first sheet. Ignored for CSV files.  
`...` Additional arguments passed to `utils::read.csv()` or `readxl::read_excel()`.

**Value**

A data frame with column names converted to snake\_case. Duplicate column names are made unique by appending numeric suffixes.

**Examples**

```
# Create a temporary CSV for demonstration
tmp <- tempfile(fileext = ".csv")
write.csv(mtcars, tmp, row.names = FALSE)
df <- import_data(tmp)
head(df)
unlink(tmp)
```

---

launch

*Launch the earthUI Shiny application*

---

**Description**

Opens an interactive 'shiny' GUI for building and exploring 'earth' (MARS-style) models. The application provides data import, variable configuration, model fitting, result visualization, and report export.

**Usage**

```
launch(port = 7878L, ...)
```

**Arguments**

port	Integer. Port number for the Shiny app. Defaults to 7878. A fixed port ensures browser localStorage (saved settings) persists across sessions.
...	Additional arguments passed to <code>shiny::runApp()</code> .

**Value**

This function does not return a value; it launches the Shiny app.

**Examples**

```
if (interactive()) {
  launch()
}
```

---

list_g_functions	List g-function groups from a fitted earth model
------------------	--

---

### Description

Returns a data frame describing each non-intercept g-function group from the model equation, including degree, factor count, graph dimensionality, and the number of terms. The g-function notation is  $^f g_k^j$  where  $f$  = number of factor variables (top-left),  $j$  = degree of interaction (top-right),  $k$  = position within the degree group (bottom-right).

### Usage

```
list_g_functions(earth_result)
```

### Arguments

`earth_result` An object of class "earthUI\_result" as returned by `fit_earth()`.

### Value

A data frame with columns:

**index** Integer. Sequential index (1-based).

**label** Character. Variable names in the group.

**g\_j** Integer. Degree of the g-function (top-right superscript).

**g\_k** Integer. Position within the degree (bottom-right subscript).

**g\_f** Integer. Number of factor variables (top-left superscript).

**d** Integer. Graph dimensionality (degree minus factor count).

**n\_terms** Integer. Number of terms in the group.

### Examples

```
result <- fit_earth(mtcars, "mpg", c("cyl", "disp", "hp", "wt"))
list_g_functions(result)
```

---

plot\_actual\_vs\_predicted  
*Plot actual vs predicted values*

---

**Description**

Creates a scatter plot of actual vs predicted values with a 1:1 reference line.

**Usage**

```
plot_actual_vs_predicted(earth_result, response_idx = NULL)
```

**Arguments**

`earth_result` An object of class "earthUI\_result" as returned by `fit_earth()`.  
`response_idx` Integer or NULL. For multivariate models, which response column to plot (1-based). Default NULL uses the first response.

**Value**

A `ggplot2::ggplot` object.

**Examples**

```
result <- fit_earth(mtcars, "mpg", c("cyl", "disp", "hp", "wt"))
plot_actual_vs_predicted(result)
```

---

plot\_contribution *Plot variable contribution*

---

**Description**

Creates a scatter plot showing each variable's actual contribution to the prediction. For each observation, the contribution is the sum of coefficient \* basis function value across all terms involving that variable.

**Usage**

```
plot_contribution(earth_result, variable, response_idx = NULL)
```

**Arguments**

`earth_result` An object of class "earthUI\_result" as returned by `fit_earth()`.  
`variable` Character string. Name of the predictor variable to plot.  
`response_idx` Integer or NULL. For multivariate models, which response column to plot (1-based). Default NULL uses the first response.

**Value**

A `ggplot2::ggplot` object.

**Examples**

```
result <- fit_earth(mtcars, "mpg", c("cyl", "disp", "hp", "wt"))
plot_contribution(result, "wt")
```

---

plot\_correlation\_matrix

*Plot correlation matrix*

---

**Description**

Creates a heatmap of pairwise correlations among the target variable and numeric predictors, with cells colored by degree of correlation and values printed in each cell.

**Usage**

```
plot_correlation_matrix(earth_result)
```

**Arguments**

`earth_result` An object of class "earthUI\_result" as returned by `fit_earth()`.

**Value**

A `ggplot2::ggplot` object.

**Examples**

```
result <- fit_earth(mtcars, "mpg", c("cyl", "disp", "hp", "wt"))
plot_correlation_matrix(result)
```

---

plot_g_contour	<i>Plot g-function as a static contour (for reports)</i>
----------------	--

---

### Description

Creates a ggplot2 visualization for any g-function group. For  $d \leq 1$ , produces a 2D scatter plot (same as `plot_g_function()`). For  $d \geq 2$ , produces a filled contour plot suitable for static formats like PDF and Word.

### Usage

```
plot_g_contour(earth_result, group_index, response_idx = NULL)
```

### Arguments

earth_result	An object of class "earthUI_result" as returned by <code>fit_earth()</code> .
group_index	Integer. Index of the g-function group (1-based, from <code>list_g_functions()</code> ).
response_idx	Integer or NULL. For multivariate models, which response column to plot (1-based). Default NULL uses the first response.

### Value

A `ggplot2::ggplot` object.

### Examples

```
result <- fit_earth(mtcars, "mpg", c("cyl", "disp", "hp", "wt"))
plot_g_contour(result, 1)
```

---

plot_g_function	<i>Plot g-function contribution</i>
-----------------	-------------------------------------

---

### Description

Creates a contribution plot for a specific g-function group. For degree-1 groups (single variable), produces a 2D scatter + piecewise-linear plot with slope labels and knot markers. For degree-2 groups (two variables), produces a 3D surface plot using `plotly` if available, or a filled contour plot.

### Usage

```
plot_g_function(earth_result, group_index, response_idx = NULL)
```

**Arguments**

earth_result	An object of class "earthUI_result" as returned by <code>fit_earth()</code> .
group_index	Integer. Index of the g-function group (1-based, from <code>list_g_functions()</code> ).
response_idx	Integer or NULL. For multivariate models, which response column to plot (1-based). Default NULL uses the first response.

**Value**

A `ggplot2::ggplot` object for  $d \leq 1$ , or a plotly widget for  $d \geq 2$  (when plotly is installed). Falls back to `ggplot2` contour if plotly is not available.

**Examples**

```
result <- fit_earth(mtcars, "mpg", c("cyl", "disp", "hp", "wt"))
plot_g_function(result, 1)
```

---

plot_g_persp	<i>Plot g-function as a static 3D perspective (for reports)</i>
--------------	---

---

**Description**

Creates a base R `persp()` 3D surface plot for g-function groups with  $d \geq 2$ . For  $d \leq 1$ , produces a 2D scatter plot (same as `plot_g_function()`). The surface is colored by contribution value using a blue-white-red scale. Suitable for PDF and Word output where interactive plotly is not available.

**Usage**

```
plot_g_persp(
  earth_result,
  group_index,
  theta = 30,
  phi = 25,
  response_idx = NULL
)
```

**Arguments**

earth_result	An object of class "earthUI_result" as returned by <code>fit_earth()</code> .
group_index	Integer. Index of the g-function group (1-based, from <code>list_g_functions()</code> ).
theta	Numeric. Azimuthal rotation angle in degrees. Default 30.
phi	Numeric. Elevation angle in degrees. Default 25.
response_idx	Integer or NULL. For multivariate models, which response column to plot (1-based). Default NULL uses the first response.

**Value**

Invisible NULL (base graphics). For  $d \leq 1$ , returns a ggplot object.

**Examples**

```
result <- fit_earth(mtcars, "mpg", c("cyl", "disp", "hp", "wt"), degree = 2L)
plot_g_persp(result, 1)
```

---

plot\_partial\_dependence

*Plot partial dependence*

---

**Description**

Creates a partial dependence plot for a selected variable from a fitted earth model.

**Usage**

```
plot_partial_dependence(  
  earth_result,  
  variable,  
  n_grid = 50L,  
  response_idx = NULL  
)
```

**Arguments**

earth_result	An object of class "earthUI_result" as returned by <code>fit_earth()</code> .
variable	Character string. Name of the predictor variable to plot.
n_grid	Integer. Number of grid points for the partial dependence calculation. Default is 50.
response_idx	Integer or NULL. For multivariate models, which response column to plot (1-based). Default NULL uses the first response.

**Value**

A `ggplot2::ggplot` object.

**Examples**

```
result <- fit_earth(mtcars, "mpg", c("cyl", "disp", "hp", "wt"))
plot_partial_dependence(result, "wt")
```

---

plot_qq	<i>Plot Q-Q plot of residuals</i>
---------	-----------------------------------

---

**Description**

Creates a normal Q-Q plot of the model residuals.

**Usage**

```
plot_qq(earth_result, response_idx = NULL)
```

**Arguments**

`earth_result` An object of class "earthUI\_result" as returned by [fit\\_earth\(\)](#).  
`response_idx` Integer or NULL. For multivariate models, which response column to plot (1-based). Default NULL uses the first response.

**Value**

A `ggplot2::ggplot` object.

**Examples**

```
result <- fit_earth(mtcars, "mpg", c("cyl", "disp", "hp", "wt"))  
plot_qq(result)
```

---

plot_residuals	<i>Plot residual diagnostics</i>
----------------	----------------------------------

---

**Description**

Creates a two-panel diagnostic plot: residuals vs fitted values and a Q-Q plot of residuals.

**Usage**

```
plot_residuals(earth_result, response_idx = NULL)
```

**Arguments**

`earth_result` An object of class "earthUI\_result" as returned by [fit\\_earth\(\)](#).  
`response_idx` Integer or NULL. For multivariate models, which response column to plot (1-based). Default NULL uses the first response.

**Value**

A `ggplot2::ggplot` object showing residuals vs fitted values. Use `plot_qq()` for the Q-Q plot separately.

**Examples**

```
result <- fit_earth(mtcars, "mpg", c("cyl", "disp", "hp", "wt"))
plot_residuals(result)
```

---

`plot_variable_importance`*Plot variable importance*

---

**Description**

Creates a horizontal bar chart of variable importance from a fitted earth model.

**Usage**

```
plot_variable_importance(earth_result, type = "nsubsets")
```

**Arguments**

`earth_result` An object of class "earthUI\_result" as returned by `fit_earth()`.  
`type` Character. Importance metric to plot: "nsubsets" (default), "gcv", or "rss".

**Value**

A `ggplot2::ggplot` object.

**Examples**

```
result <- fit_earth(mtcars, "mpg", c("cyl", "disp", "hp", "wt"))
plot_variable_importance(result)
```

---

render_report	<i>Render an earth model report</i>
---------------	-------------------------------------

---

**Description**

Renders a parameterized 'Quarto' report from the fitted 'earth' model results. Requires the 'quarto' R package and a 'Quarto' installation.

**Usage**

```
render_report(earth_result, output_format = "html", output_file = NULL)
```

**Arguments**

`earth_result` An object of class "earthUI\_result" as returned by `fit_earth()`.  
`output_format` Character. Output format: "html", "pdf", or "docx". Default is "html".  
`output_file` Character. Path for the output file. If NULL, a temporary file is created.

**Value**

The path to the rendered output file (invisibly).

**Examples**

```
result <- fit_earth(mtcars, "mpg", c("cyl", "disp", "hp", "wt"))
render_report(result, output_format = "html",
              output_file = tempfile(fileext = ".html"))
```

---

validate_types	<i>Validate declared column types against actual data</i>
----------------	---

---

**Description**

Checks each selected predictor's actual data against the user-declared type. Returns a list of errors (blocking), warnings (non-blocking), and any Date/POSIXct columns that will be auto-converted to numeric.

**Usage**

```
validate_types(df, type_map, predictors)
```

**Arguments**

<code>df</code>	A data frame.
<code>type_map</code>	Named list or character vector. Names are column names, values are declared types (e.g., "numeric", "Date").
<code>predictors</code>	Character vector of selected predictor column names.

**Value**

A list with components:

**ok** Logical. TRUE if no blocking errors found.

**warnings** Character vector of non-blocking warnings.

**errors** Character vector of blocking errors.

**date\_columns** Character vector of Date/POSIXct predictor columns that will be auto-converted to numeric.

**Examples**

```
df <- data.frame(price = c(100, 200, 300), city = c("A", "B", "C"))
types <- list(price = "numeric", city = "character")
validate_types(df, types, predictors = c("price", "city"))
```

# Index

build\_allowed\_function, 2  
build\_allowed\_function(), 6  
build\_allowed\_matrix, 3  
build\_allowed\_matrix(), 2

detect\_categoricals, 4  
detect\_types, 4

earth::earth(), 2, 3, 5, 8  
earth::evimp(), 11

fit\_earth, 5  
fit\_earth(), 8–11, 13–21  
format\_anova, 8  
format\_model\_equation, 9  
format\_summary, 10  
format\_variable\_importance, 11

ggplot2::ggplot, 14–20

import\_data, 11

launch, 12  
list\_g\_functions, 13  
list\_g\_functions(), 16, 17

plot\_actual\_vs\_predicted, 14  
plot\_contribution, 14  
plot\_correlation\_matrix, 15  
plot\_g\_contour, 16  
plot\_g\_function, 16  
plot\_g\_function(), 16, 17  
plot\_g\_persp, 17  
plot\_partial\_dependence, 18  
plot\_qq, 19  
plot\_qq(), 20  
plot\_residuals, 19  
plot\_variable\_importance, 20

readxl::read\_excel(), 11  
render\_report, 21

shiny::runApp(), 12  
utils::read.csv(), 11  
validate\_types, 21